OPERATOR GUIDE TO SERIAL I/O CAPABILITIES OF DATA I/O PROGRAMMERS

TRANSLATION-FORMAT PACKAGE 055-1901

REV C OCT 80

055-1901

TABLE OF CONTENTS

SECTION 1. INTRODUCTION

SECTION 2. TRANSLATION-FORMAT SPECIFICATIONS

SECTION 3. DATA-TRANSFER OPERATIONS

LIST OF FIGURES

LIST OF TABLES

SECTION 1.
INTRODUCTION

1.1 GENERAL

This manual defines the data-translation formats available for input and output by Data I/O's System 19 Universal Programmer.

The Data I/O System 19 is capable of interfacing with all RS232 serial equipment which employs a translation format described in this manual. The three transmission operations possible with translation formats are Input, Output, and Input Compare. These operations are accessed at the programmer's keyboard by unique "Select Codes." The procedure for implementing the operations is listed in Section 3 of this manual.

Each translation format is also assigned a Select Code, which is used to set up the programmer to send or receive data in that format. Those Select Codes are listed in Table 2-1.

1.2 SUM-CHECK FIELD

The System 19 calculates a sum-check of all data sent to or from the programmer.

A sum-check is a sixteen-bit summation of the series of data words transferred, represented as four hex digits. Carry from the sixteenth bit is discarded. See the example in Figure 1-1.

At the end of a successful Input or Input Compare operation, the programmer displays the sum-check of all data transferred. The programmer will also compare any received sum-check fields with its own calculation. If the two agree, the programmer will display the sum-check; a mismatch will produce an error message.

Output data is always followed by a sum-check field, which may be used to check subsequent input operations.

SUM-CHECK EXAMPLE:

HEX DATA	BINARY DATA
84	10000100
C1	11000001
62	01100010
24	00100100
Σ 01CB	Σ 0000 0001 1100 1011
Sum-check in hexadecimal notation	Sixteen-bit binary sum-check

Figure 1-1. Calculation of the Sum-Check of Four Eight-Bit Bytes (Hex): 84, C1, 62 and 24

1.3 ADDRESS SETTING

Address setting differs between formats with expressed address fields and formats without. In formats without address fields, the Begin RAM Address (BLOCK LIMIT L1) defines the first RAM address at which data will be transferred. In formats with expressed address fields, the address offset can also be used to define this address. Specifications for the Begin RAM Address and the address offset are given in paragraph 3.2.1.

SECTION 2
TRANSLATION-FORMAT SPECIFICATIONS

This section gives the specifications of the translation formats available for input and output by the System 19. The Select Codes for each of the formats are listed in Table 2-1.

Table 2-1. Select Codes for Translation Formats

FORMAT	SELECT CODE
Binary	10
DEC Binary	11
ASCII-BNPF	01 (05)*
ASCII-BHLF	02 (06)*
ASCII-B1ØF	03 (07)*
5-level BNPF	08 (09)*
Spectrum	12 (13)*
ASCII-Octal (Space)	30 (35)†
ASCII-Octal (Percent)	31 (36)†
ASCII-Octal (Apostrophe)	32
ASCII-Octal SMS	37
ASCII-Hex (Space)	50 (55)†
ASCII-Hex (Percent)	51 (56)†
ASCII-Hex (Apostrophe)	52
ASCII-Hex SMS	57
ASCII-Hex (Comma)	53 (58)†
RCA Cosmac	70
Fairchild Fairbug	80
MOS Technology	81
Motorola Exorciser	82
Intel Intellec 8/MDS	83 ✓
Signetics Absolute Object	85
Tektronix Hexadecimal	86 ✓

* For transmission of data without start codes, these alternate Select Codes are used.

† For transmission of data with the SOH (CTRL A) start code, these alternate Select Codes are used.

2.1 BINARY TRANSFER, Select Code 10

Data transfer in the Binary format consists of a stream of 8-bit data words preceded by a byte count and followed by a sum-check. The Binary format does not have addresses.

A tape generated by the programmer will contain a 5-byte, arrow-shaped header followed by a null and a 4-nibble byte count. The start code, a nonprintable rubout in even parity, follows the byte count. The end of data is signalled by two nulls and a 2-byte sum-check of the data field.

2.1.1 INPUT REQUIREMENTS

The programmer stores incoming binary data upon receipt of the start character. Data is stored in RAM sequentially starting at the Begin RAM Address (L1) and ending at the last incoming data byte. Transmission may be aborted by pressing the KEYBD key on the System 19.

For binary tapes with no byte count or sum-check, the number of bytes to be transmitted (L2) must be set, in order to indicate the end of transmission.

2.1.2 OUTPUT CHARACTERISTICS

Hard copy from binary files is not useful because the terminal interprets output from the programmer as ASCII characters. Output can be used to produce binary tapes.

Programmer output in binary consists of a sequential data stream preceded by the arrow-shaped header, one null, byte count and start code and followed by 2 nulls and a sum-check. As with Input, transmission may be aborted by pressing the KEYBD key on the System 19.

NOTE

If four-bit operation is selected, data will be output in 8-bit form, but the high order bytes will be zeros.

2.2 DEC BINARY FORMAT, Select Code 11

Data transmission in the DEC Binary format is a stream of 8-bit data words with no control characters except the start code. The number of bytes to be transmitted (L2) must be set, in order to indicate the end of transmission.

The start code is one null preceded by at least one rubout. A tape output from the programmer will contain 32 rubouts in the leader. The DEC Binary format does not have addresses.

NOTE

If four-bit operation is selected, data will be output in 8-bit form, but the high order bytes will be zeros.

2.3 ASCII BINARY FORMATS, Select Codes 01, 02 and 03 (or 05, 06 and 07)

In these formats, bytes are recorded in ASCII codes with binary digits represented by "N's" and "P's", "L's" and "H's," and "1's" and "0's" respectively. The ASCII Binary Formats do not have addresses.

Data is transferred in 8-bit form unless 4-bit form is specified, or unless a 4-bit socket adapter is installed.

2.3.1 INPUT REQUIREMENTS

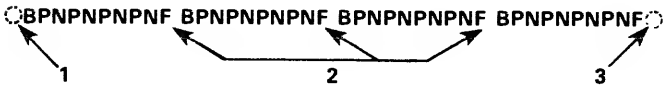
Figure 2-1 shows four data bytes coded in each of the three ASCII Binary formats. Incoming bytes are stored in RAM sequentially starting at the Begin RAM Address. Bytes are sandwiched between "B" and "F" characters and are normally separated by spaces. Any other characters, such as carriage returns or line feeds, may be inserted between an "F" and the next "B". The start code is a nonprintable STX, control B (or hex Ø2 in "no parity"), and the end code is a nonprintable ETX, control C (or hex Ø3 in "no parity").

NOTE

Data without a start code may be input to or output from the programmer by use of alternate Format Select Codes. These are: ASCII-BNPF, 05; ASCII-BHLF, 06; ASCII-B10F, 07.

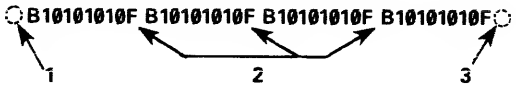
A single data byte can be aborted if the programmer receives an E character between B and F characters. Data will continue to be stored in sequential RAM addresses. The entire data transfer can be aborted by pressing the KEYBD key on the programmer.

ASCII-BNPF, Select Code 01 (or 05)



ASCII-BHLF, Select Code 02 (or 06)

ASCII-B10F, Select Code 03 (or 07)



NOTES

1. **Start code is a nonprintable STX — CTRL B (start code is optional).**
2. **Characters such as spaces, carriage returns and line feeds may appear between bytes.**
3. **End code is a nonprintable ETX — CTRL C.**
4. **Data can also be expressed in 4-bit words**

Figure 2-1. ASCII Binary Formats

2.3.2 OUTPUT CHARACTERISTICS

Data is output in four-byte lines with a space between bytes. The programmer begins with an STX (optionally) and ends with an ETX. The programmer sends data beginning at the defined Begin RAM Address (L1)and continuing up to the end of RAM or until the specified number of bytes (L2) has been sent. The transmission is preceded and followed by fifty null characters.

2.4 5-LEVEL BNPF FORMAT, Select Code 08 (or 09)

Except for the start and end codes, the same character set and specifications are used for the ASCII-BNPF and 5-level BNPF Formats.

Data for input to the programmer is punched on 5-hole Telex paper tapes to be read by an ASCII-based reader that has an adjustable tape guide. The reader reads the tape as it would an 8-level tape, recording the five holes that are on the tape as five bits of data. The three most significant bits are recorded as if they were holes on an 8-level tape. The programmer's software converts the resulting eight-bit codes into valid data for entry in RAM.

The start code for the format is a left parenthesis, ("Figs K" on a Telex machine), and the end code is a right parenthesis, ("Figs L" on a Telex machine). The 5-level BNPF Format does not have addresses.

NOTE

Data without a start code may be input to or output from the programmer by use of the alternate Format Select Code, 09.

2.5 SPECTRUM FORMAT, Select Code 12 (or 13)

In this format, bytes are recorded in ASCII codes with binary digits represented by "1's" and "0's".

Data is transferred in either 4- or 8-bit form. Each byte is preceded by an address.

NOTE

The Spectrum format does not recognize address offsets.

2.5.1 INPUT REQUIREMENTS

Figure 2-2 shows two data bytes coded in the Spectrum format. Incoming bytes are stored in RAM sequentially starting at the Begin RAM Address. Bytes are sandwiched between the space and carriage-return characters and are normally separated by line feeds. The start code is a nonprintable STX, control B (or hex Ø2 with no parity), and the end code is a nonprintable ETX, control C (or hex Ø3 with no parity).

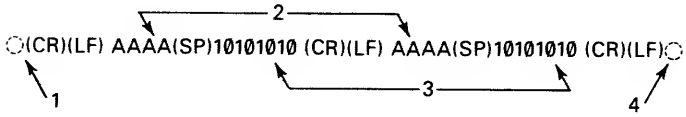
NOTE

Data without a start code may be input to or output from the programmer by use of the alternate Format Select Code (13).

A single data byte can be aborted if the programmer receives an "E" character between a space and a carriage return. Data will continue to be stored in sequential RAM addresses. The entire data transfer can be aborted by pressing the KEYBD key on the programmer.

2.5.2 OUTPUT CHARACTERISTICS

Data output to a printer will have one address and one byte of data on each line. The programmer first sends an STX (optionally), then all data in the defined block (L2) starting at the Begin RAM Address (L1), and finally an ETX. The transmission is preceded and followed by 50 null characters.



NOTES

1. **Start code is a nonprintable STX (start code is optional).**
2. **Address code is 4 hex digits.**
3. **4 or 8 data bits appear between space and carriage return.**
4. **End code is a nonprintable ETX.**

Figure 2-2. Spectrum Format

2.6 ASCII OCTAL AND HEX FORMATS, Select Codes 30-37 and 50-58

Each of these formats has a start and end code and similar address and sum-check specifications.

NOTE

ASCII-Octal and -Hex (Space) and (Comma) tapes generated on the Data I/O Model 5 Programmer have the start code "SOH" (Control A). To input or output a tape with the SOH start code, use the alternate Select Codes shown in Figure 2-3.

2.6.1 INPUT REQUIREMENTS

Figure 2-3 illustrates four data bytes coded in each of the nine ASCII-Octal and Hex Formats. Data in these formats is organized in sequential bytes separated by the execute character (space, percent, apostrophe or comma). Characters immediately preceding the execute character are interpreted as data. ASCII-Hex and Octal formats can express either 8- or 4-bit data. 8-bit data is expressed by 3 octal or 2 hex characters; 4-bit data, by 2 octal or 1 hex character. Line feeds, carriage returns and other characters may be included in the data stream as long as a data byte directly precedes each execute character.

Although each data byte has an address, most addresses are unexpressed. Data bytes are addressed sequentially unless an explicit address is included in the data stream. This address is preceded by a "$" and an "A", must contain 2 to 4 hex or 3 to 6 octal characters, and must be followed by a comma, except for the ASCII-Hex (Comma) Format, which uses a period. The programmer skips to the new address to store the next data byte; succeeding bytes are again stored sequentially. See Figure 2-4.

Each format has an end code, which terminates input operations. However, if a new start code follows within 16 characters of an end code, input will continue uninterrupted.

NOTE

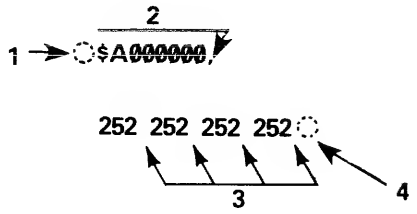
To avoid a time-out error, at least sixteen characters must follow an end code or time-out must be disabled. (See Table 3-3, System 19 Universal Programmer Manual.)

After receiving the final end code following an input operation, the programmer calculates a sum-check of all incoming data. Optionally, a sum-check can also be entered in the input data stream. The programmer compares this sum-check field with its own calculated sum-check. If they match, the programmer will simply display the sum-check; if they do not match, the programmer will display a sum-check error. Specifications for the optional sum-check field are given in Figure 2-5.

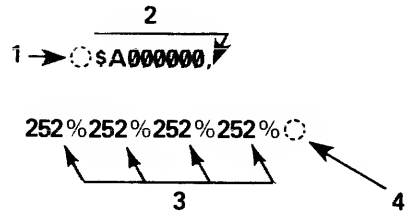
NOTE

ASCII-Octal and Hex Formats can also be expressed in 4-bit form.

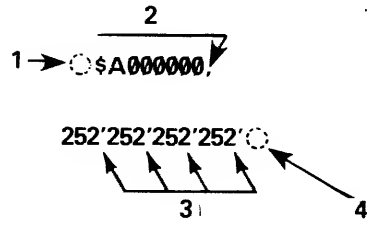
ASCII-Octal (Space), Select Code 30 (or 35)



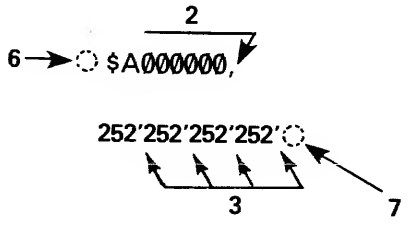
ASCII-Octal (Percent), Select Code 31 (or 36)



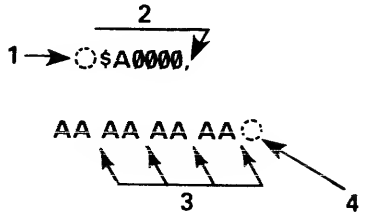
ASCII-Octal (Apostrophe), Select Code 32



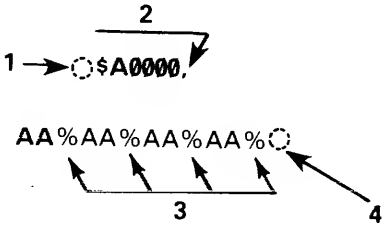
ASCII-Octal SMS, Select Code 37



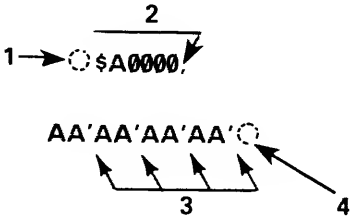
ASCII-Hex (Space), Select Code 50 (or 55)



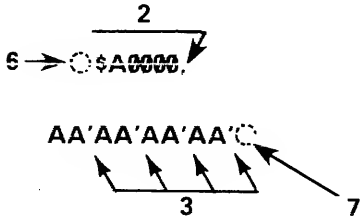
ASCII-Hex (Percent), Select Code 51 (or 56)



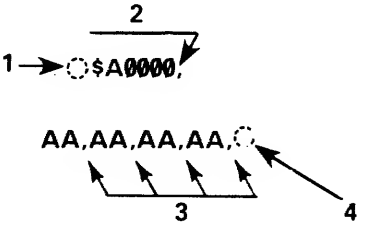
ASCII-Hex (Apostrophe), Select Code 52



ASCII-Hex SMS, Select Code 57



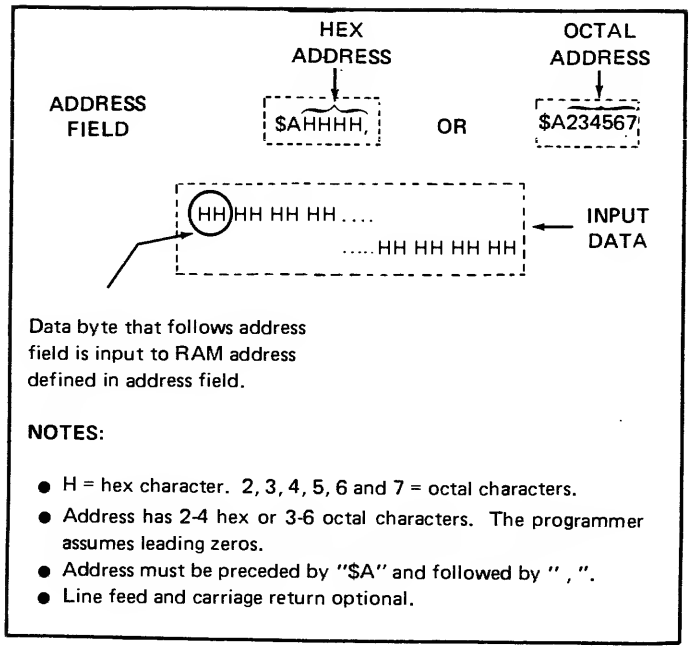
ASCII-Hex (Comma), Select Code 53 (or 58)



NOTES:

1. Start code is nonprintable STX — CTRL B (optionally SOH — CTRL A).
2. Optional address code may precede any data byte. Up to six address digits in octal formats, four in hex.
3. Execute code.
4. End code is a nonprintable ETX — CTRL C.
5. Data can also be expressed in 4-bit form.
6. Start code is nonprintable SOM — CTRL R.
7. End code is a nonprintable EOM — CTRL T.

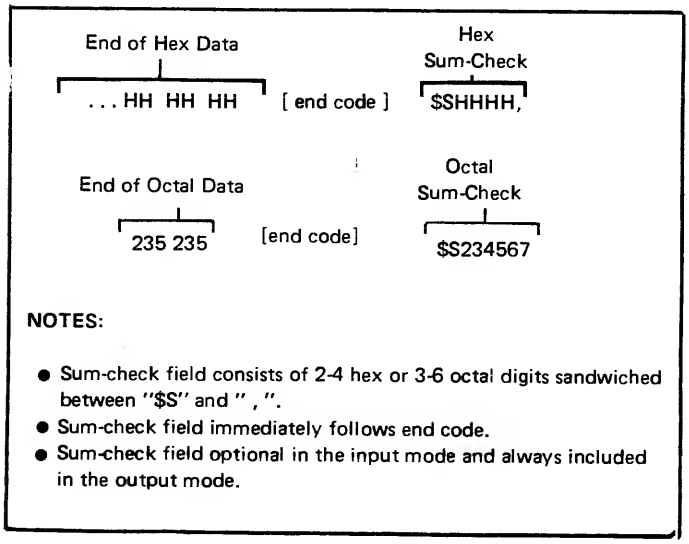
Figure 2-3. ASCII-Octal and Hex Formats



NOTES:

- H = hex character. 2, 3, 4, 5, 6 and 7 = octal characters.
- Address has 2-4 hex or 3-6 octal characters. The programmer assumes leading zeros.
- Address must be preceded by "$A" and followed by " , ".
- Line feed and carriage return optional.

Figure 2-4. Optional Address Field in ASCII-Hex and Octal Formats



NOTES:

- Sum-check field consists of 2-4 hex or 3-6 octal digits sandwiched between "$S" and " , ".
- Sum-check field immediately follows end code.
- Sum-check field optional in the input mode and always included in the output mode.

Figure 2-5. Syntax of the Sum-Check Field in I/O Operations

2.6.2 OUTPUT CHARACTERISTICS

Data is output from the programmer between the address limits set by the Begin RAM Address and the number of bytes to be transferred. After these parameters have been set, output is begun by invoking the Output command. The programmer divides output data into eight-line blocks.

Data transmission is begun with the start code, a nonprintable STX (optionally SOH)*. Data blocks follow, each one prefaced by an address for the first data byte in the block. The end of transmission is signaled by the end code, a nonprintable ETX. Directly following the end code is a sum-check of the transferred data. The transmission is preceded and followed by fifty null characters.

2.7 RCA COSMAC FORMAT, Select Code 70

2.7.1 INPUT REQUIREMENTS

Data in this format begins with a start record consisting of the start character (!M or ?M), an address field, and a space. See Figure 2-6.

The start character ?M is sent to the programmer only by the development system. This happens when the operator enters the interrogation ?M at a terminal (linked in parallel with the programmer to the development system), followed by the address in the development system memory where data transmission is to begin, followed by the number of bytes to be transferred, followed by a carriage return. The development system responds by sending ?M to the programmer, followed by the starting address, followed by a data stream which conforms to the data input format described below. Transmission stops when the specified number of bytes has been transmitted.

Address specification is required for only the first data byte in the transfer. An address must have 1 to 4 hex characters and be followed by a space. The programmer records the next hex character after the space as the start of the first data byte. (A carriage return must follow the space if the start code ?M is used.) Succeeding bytes are recorded sequentially.

Each data record is followed by a comma (,) if the next record is not preceded by an address, or by a semicolon (;) if the next record starts with an address. Records consist of data bytes expressed as two hexadecimal characters and followed by either a comma or semicolon and a carriage return. Any characters received between a comma or semicolon and a carriage return will be ignored by the programmer.

The carriage-return character is significant to this format because it can signal either the continuation or the end of data flow: if the carriage return is preceded by a comma or semicolon, more data must follow; the absence of a comma or a semicolon before the carriage return indicates the end of transmission.

2.7.2 OUTPUT CHARACTERISTICS

Output data records are followed by either a comma or semicolon and a carriage return.

The Start-of-File and End-of-File Records are expressed exactly as in Input.

The transmission is preceded and followed by fifty null characters.

* ASCII-Octal SMS and ASCII-Hex SMS use SOM (CTRL R) as a start code and EOM (CTRL T) as an end code.

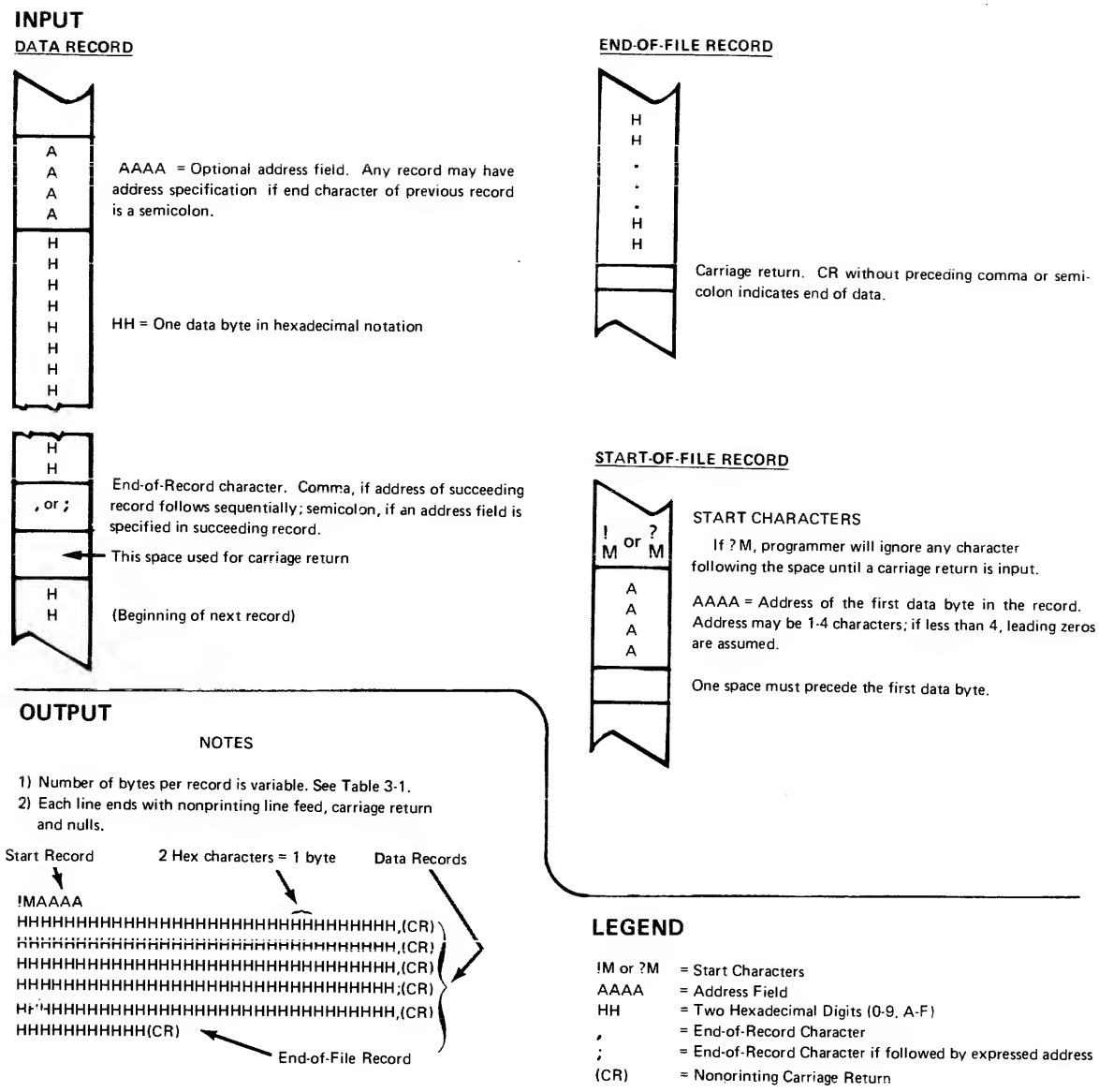


Figure 2-6. Specifications for Valid RCA Cosmac Data Files

2.8 MICROPROCESSOR FORMATS

Data in these formats is organized into records characterized by expressed addresses and error-check codes. Each format has record start characters and checksums. Records are independent; that is, the programmer can accept addresses in nonsequential order. (The Fairchild Fairbug format differs from the other microprocessor formats in address setting. See the format description in paragraph 2.8.1.)

2.8.1 INPUT REQUIREMENTS

Fairchild Fairbug, Select Code 80

In the Fairbug format, input and output requirements are identical; both have eight-byte records and identical control characters.

Figure 2-7 simulates a valid Fairchild data file. A file begins with a five-character prefix and ends with a one-character suffix. The Start-of-File character is an "S", followed by the address of the first data byte. Each data byte is represented by two hexadecimal characters.

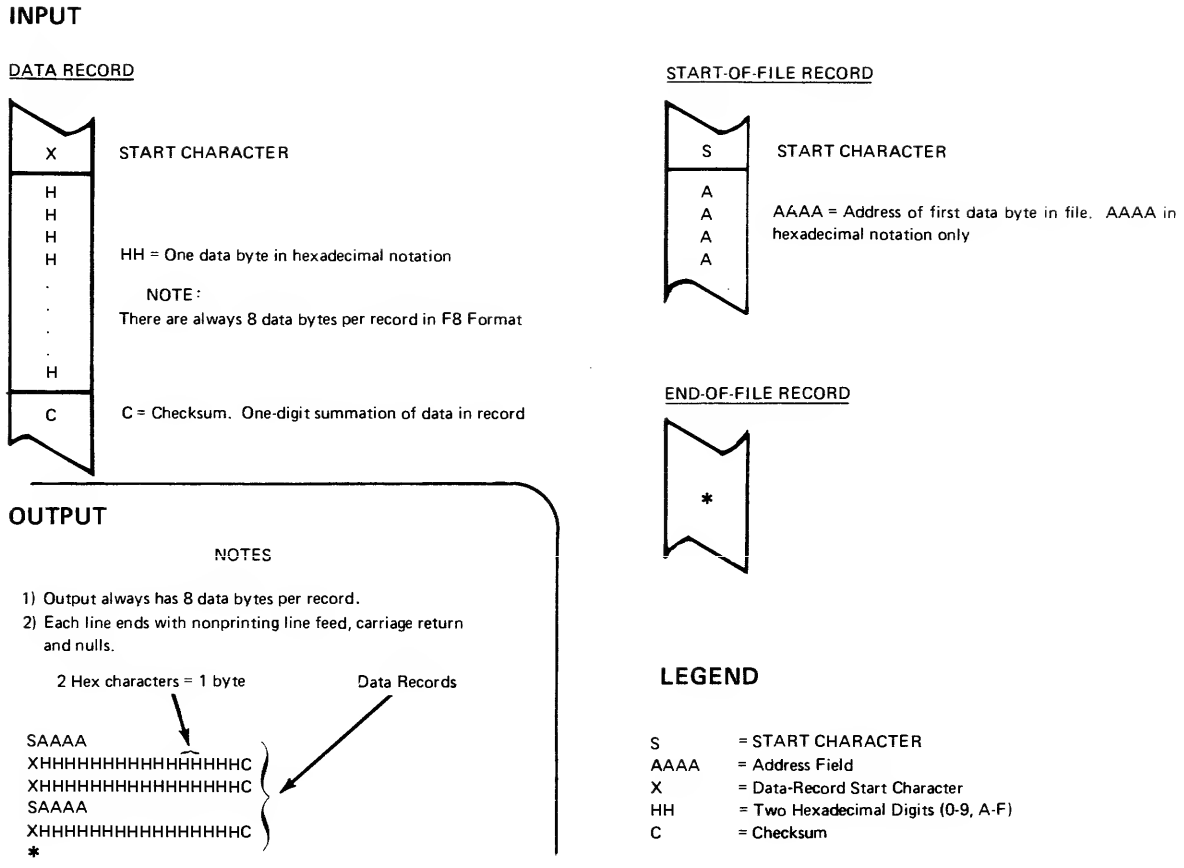


Figure 2-7. Specifications for Valid Fairchild Fairbug Data Files

NOTE

Address specification is optional in this format; a record with no address directly follows the previous record.

A one-digit hexadecimal checksum follows the data in each data record. The checksum represents, in hexadecimal notation, the sum of the binary equivalents of the sixteen digits in the record; the half carry from the fourth bit is ignored.

The programmer ignores any character (except for address characters) between a checksum and the start character of the next data record. These spaces can be used for any comments.

The last record consists only of an asterisk (*), which indicates the end of data transmission.

MOS Technology Format, Select Code 81

The data in each record is sandwiched between a seven-character prefix and a four-character suffix. The number of data bytes in each record must be indicated by

the byte count in the prefix. The input file can be divided into records of various lengths.

Figure 2-8 simulates a series of valid data records. Each data record begins with a semicolon (;). The programmer will ignore all characters received prior to the first semicolon. All other characters in a valid record must be valid hex digits (0-9, A-F). A two-digit byte count follows the start character; the byte count, expressed in hexadecimal digits, must equal the number of data bytes in the record. The next four digits make up the address of the first data type in the record. Data bytes follow, each represented by two hexadecimal digits.

The suffix is a four-character checksum, which represents the sixteen-bit binary sum of all hexadecimal bytes in the record, including the address and byte count. Carry from the most significant bit is dropped.

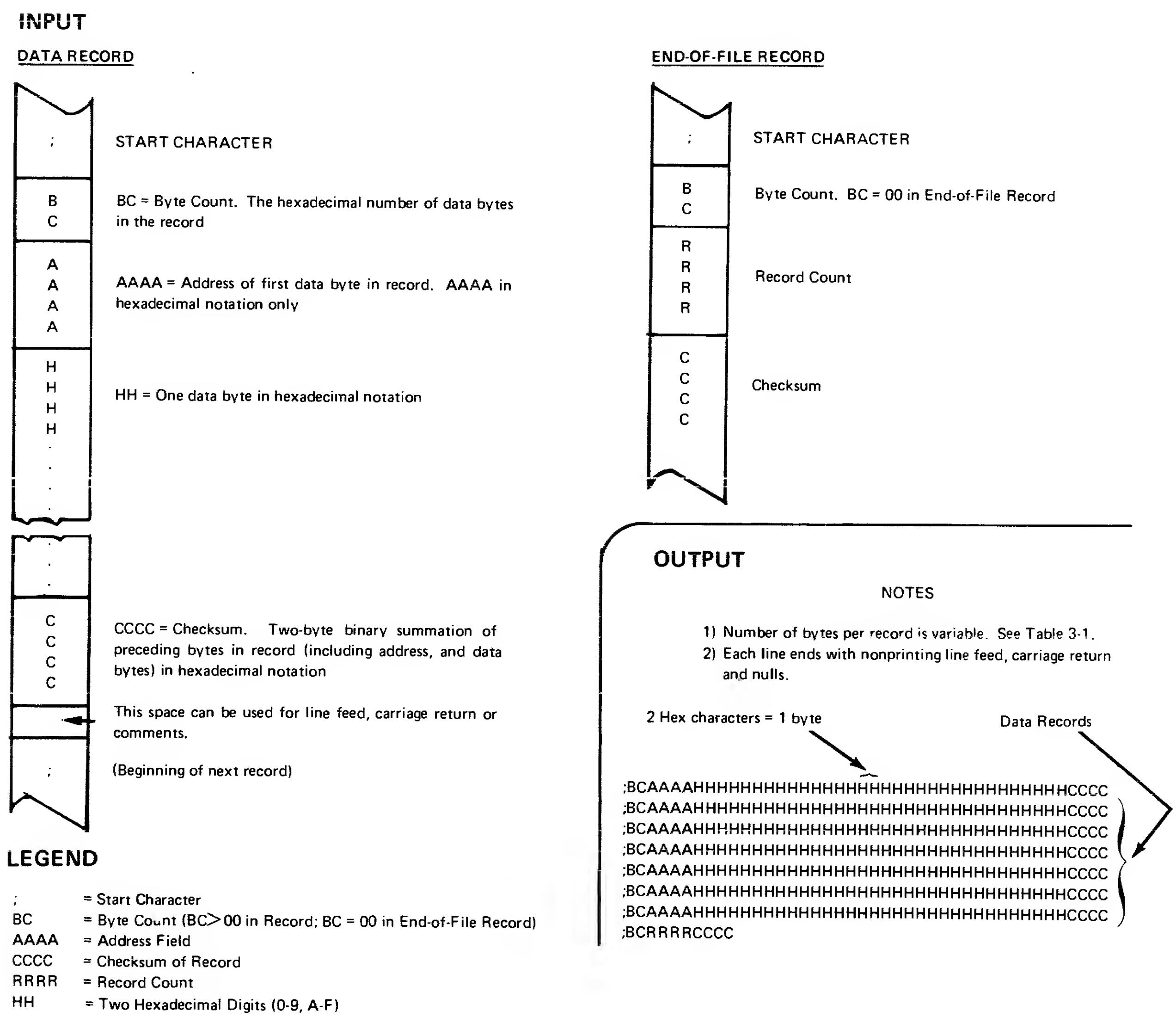


Figure 2-8. Specifications for Valid MOS Technology Data Files

Motorola Exorciser Format, Select Code 82

Motorola data files may begin with a sign-on record, which is initiated by the code SØ. Valid data records start with an eight-character prefix and end with a two-character suffix. Figure 2-9 demonstrates a series of valid Motorola data records.

Each data record begins with the start characters, "S1"; the programmer will ignore all earlier characters. The third and fourth characters represent the byte count, which expresses the number of data, address and checksum bytes in the record. The address of the first data byte in the record is expressed by the last four characters of the prefix. Data bytes follow, each represented by two hexadecimal characters. The number of data bytes occurring must be three less than the byte count. The suffix is a two-character checksum.

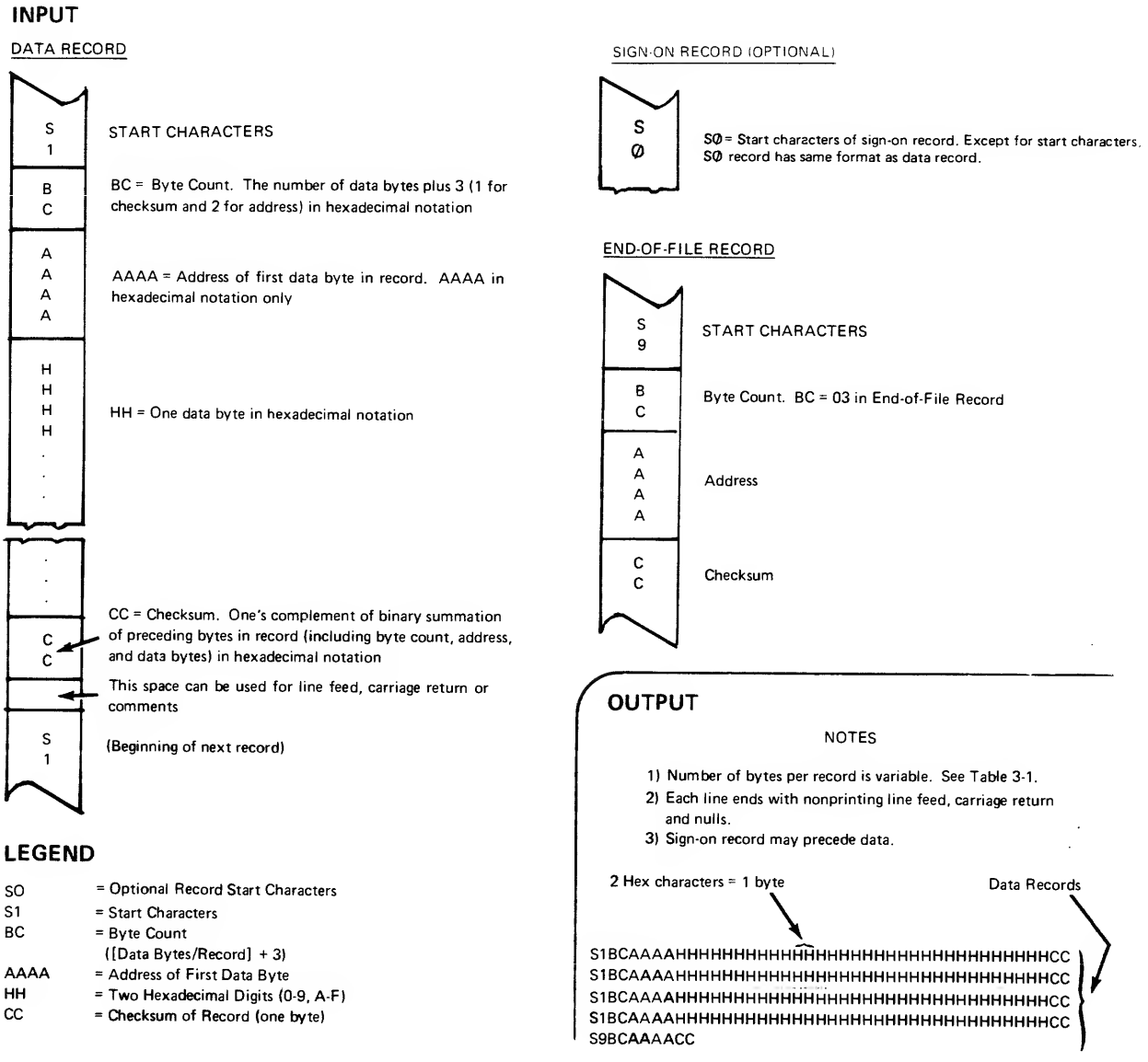


Figure 2-9. Specifications for Valid Motorola Data Files

Intel Intellec 8/MDS Format, Select Code 83

Intel data records begin with a nine-character prefix and end with a two-character suffix. The byte count must equal the number of data bytes in the record.

Figure 2-10 simulates a series of valid data records. Each record begins with a colon (:), which is followed by a two-character byte count. The four digits following the byte count give the address of the first data byte.

Each data byte is represented by two hex digits; the number of data bytes in each record must equal the byte count.

The suffix is a two-digit checksum, the two's complement of the binary summation of the previous bytes in the record.

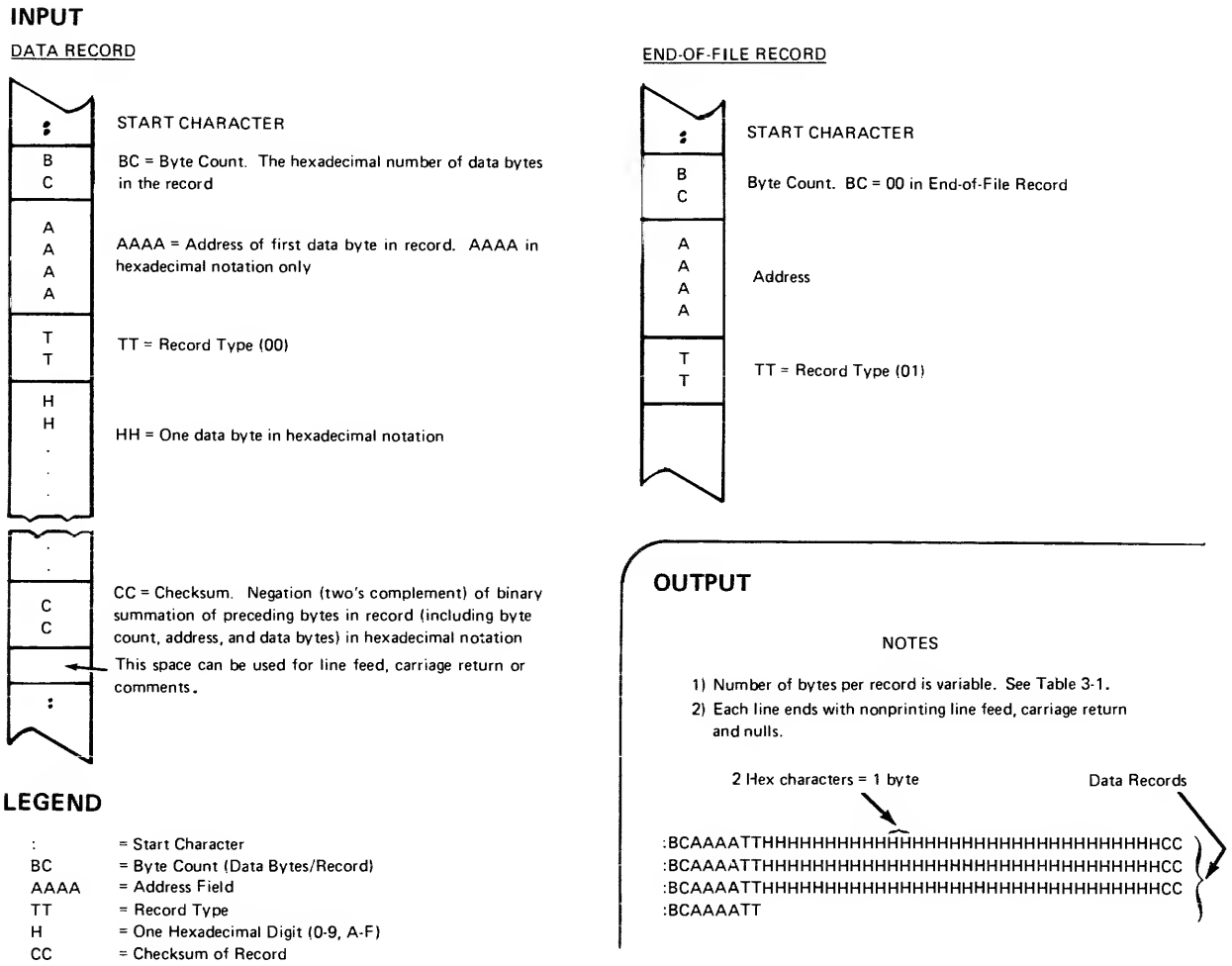


Figure 2-10. Specifications for Valid Intel Intellec 8/MDS Data Files

Signetics Absolute Object Format, Select Code 85

Figure 2-11 shows the specifications of Signetics format files. The data in each record is sandwiched between a nine-character prefix and a two-character suffix.

The start character is a colon (:). This is followed by the address of the first data byte, the byte count, and a two-digit address check. Data is represented by pairs of hexadecimal characters. The byte count must equal the number of data bytes in the record. The suffix is a two-character data check, calculated using the same operations described in Figure 2-11 for the address check.

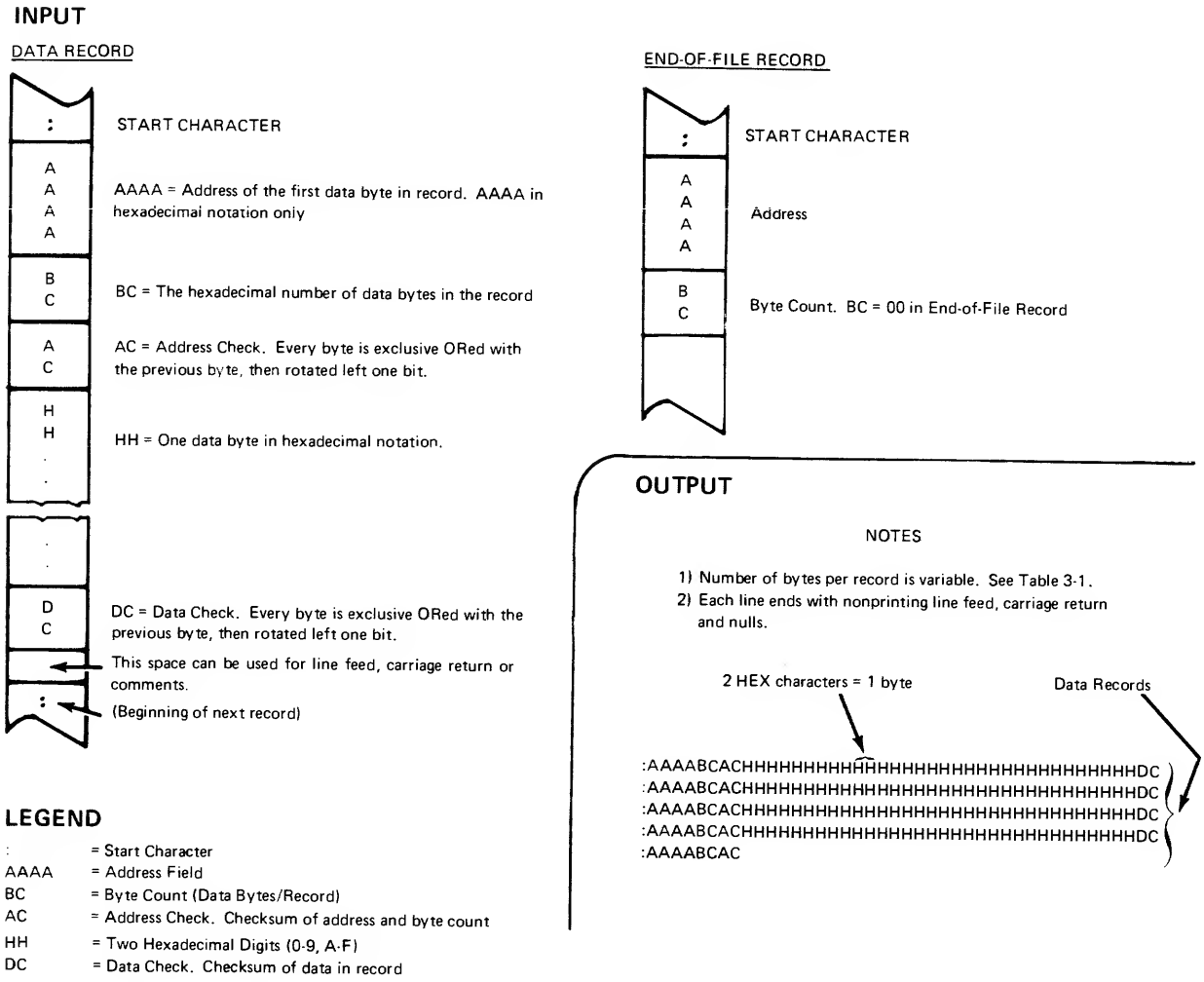


Figure 2-11. Specifications for Valid Signetics Absolute Object Data Files

Tektronix Hexadecimal Format, Select Code 86

Figure 2-12 illustrates a valid Tektronix data file. The data in each record is sandwiched between the start character, a slash (/), and a two-character checksum. Following the start character, the next four characters of the prefix express the address of the first data byte. This address is followed by a byte count, which represents the number of data bytes in the record, and by a checksum of the address and byte count. Data bytes follow, represented by pairs of hexadecimal characters and succeeded by a checksum of the data bytes. The End-of-File record consists only of control characters used to signal the end of transmission, and a byte count and checksum for verification.

If the programmer receives an abort record it will ignore it. An abort contains a start character of two slashes (//), error information and a carriage return.

2.8.2 OUTPUT CHARACTERISTICS

Data is output from the programmer, starting at the Begin RAM Address (L1), and continuing until the number of bytes in the specified block (L2) has been transmitted. Block Limit L3 is not used.

After the BLOCK LIMITS have been set, output is begun by invoking the Output command.

The programmer divides output data into records prefaced by a start character and by an address field for the first data byte in the record. The transmission is preceded and followed by fifty null characters.

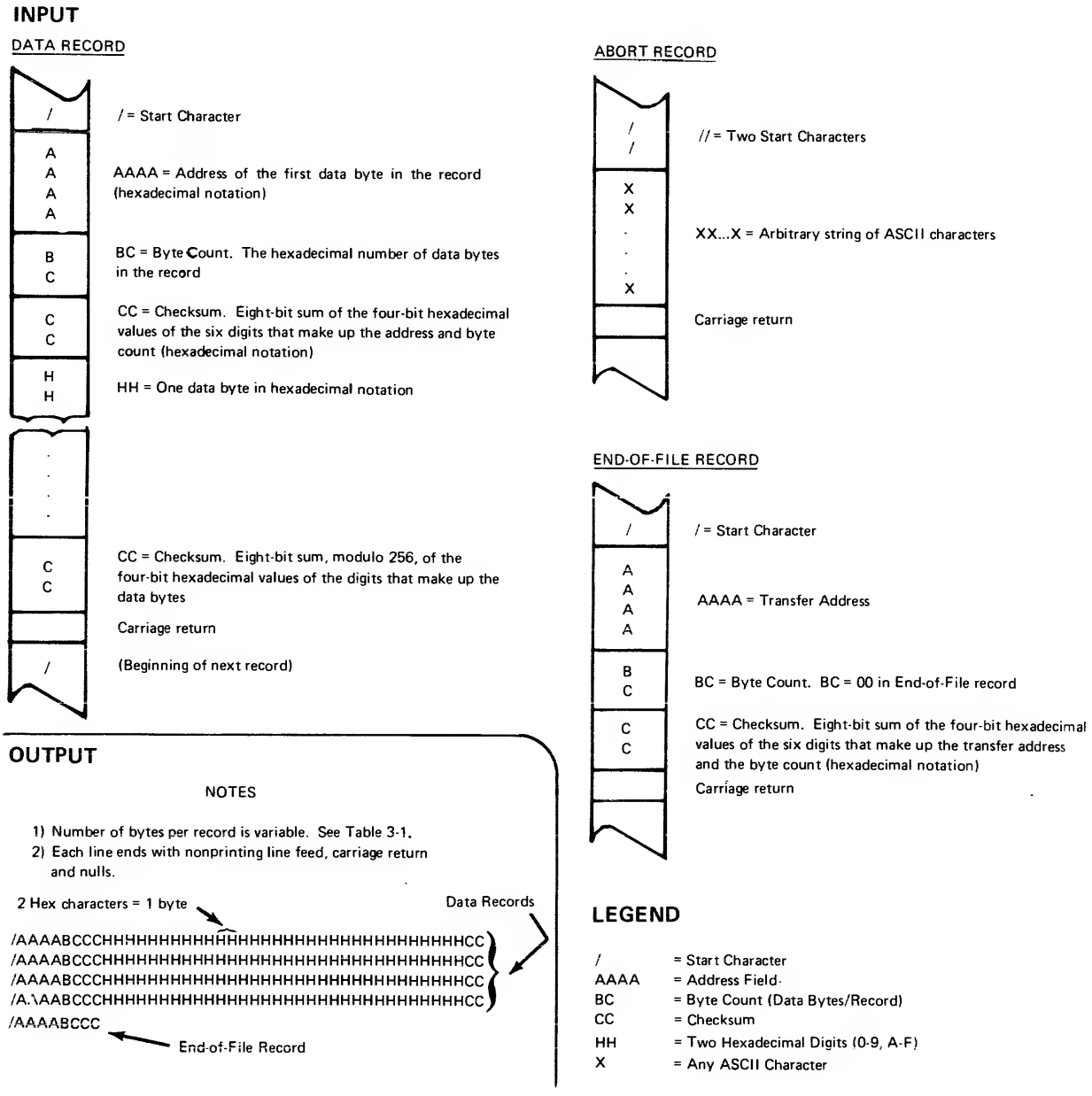


Figure 2-12. Specifications for Valid Tektronix Hexadecimal Data Files

SECTION 3.
DATA-TRANSFER OPERATIONS

3.1 PREPARATION

For successful data transfer, several factors must be consistent between the sending and receiving machines. These are covered in paragraphs 3.1.1 — 3.1.5.

Additionally, several Select Code commands are employed exclusively with I/O operations in the various formats. These commands, listed in Table 3-1, are integral to successful data-transfer operations.

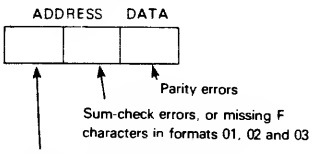
3.1.1 POLARITY

In both Input and Output operations, data sent between devices or peripherals and RAM is transmitted only in normal (high-level active) logic. However, data stored in RAM may be changed to inverted logic for use in that form; the procedure for inverting logic is the Complement Memory Command, described in Section 3 of the System 19 Operation and Maintenance Manual.

3.1.2 COMPATIBILITY OF PARITY, STOP BITS AND BAUD RATE

For data transmission between a remote device and the programmer, the machines must be set with compatible parity, stop bits and baud rate. For these settings and for proper cable connections, refer to Section 2 of the System 19 O & M Manual.

Table 3-1. Select Codes for Operations

CODE	NAME	PARAMETER (Default Value)	DESCRIPTION
B3	Format Status	–	Programmer displays three 2-digit numbers. From left to right: 1. The select Code of the format in effect 2. The hex number of nulls selected 3. The hex number of bytes per output record selected.
B4	Translator Error Status	–	Displays up to three two-digit numbers which represent the total number of errors in a data transfer. The relevance of the numbers in the address display depends on the format. The data display always shows the number of parity errors.  Non-hex characters (Formats 80-86) Missing data characters (Formats 12, 13, 30-37 and 50-58) Non-data characters (Formats 01-03)
B5	Input Compare Error Count	–	Displays the decimal number of input compare errors.
B6	Input-Buffer Overflow Errors	–	Displays the decimal number of characters received by the serial-input port buffer after the programmer sent a stop signal to the sending instrument.
D0	Output (With Remote Control)	Address Offset (0)	Prepares the programmer to output data on receipt of X-ON character, DC1 (Control Q), from remote instrument. Programmer will halt operation when remote instrument sends X-OFF character, DC3 (Control S).
D1	Input	Address Offset (First Input Address)	Initiates the input to RAM via the serial port.
D2	Input (With Instrument Control)	Address Offset (First Input Address)	Initiates the input of data to RAM via the serial port. The machine outputs the X-ON character, DC1 (Control Q), to start the remote instrument, and outputs the X-OFF character, DC3 (Control S), to stop the instrument.
D3	Compare	Address Offset (First Remote Address)	Initiates comparison of RAM data with data presented at the serial port.
D4	Compare (With Instrument Control)	Address Offset (First Remote Address)	Initiates the comparison of RAM data with data presented at the serial port. The machine outputs the X-ON character, DC 1 (Control Q), to start the remote instrument, and outputs the X-OFF character, DC3 (Control S), to stop the instrument.
D5	Output	Address Offset (0)	Initiates data output from the programmer.
D6	Output (With Instrument Control)	Address Offset (0)	Initiates data output from the programmer. The programmer output the punch-on character, DC2 (Control R), prior to data transfer and the punch-off character, DC4 (Control T), on completion.
D7	Output Nulls	–	Sends 50 nulls from the serial port.
D8	Select Record Size	–	Changes the number of bytes per output record in formats with variable record lengths. The keyed-in number must be in hex notation. Default value: 10 hex.
D9	Set Nulls	Number of Nulls (1)	Allows selection (in hexadecimal notation) of up to 254 nulls (FE) following each data record. Selecting 255 (Hex FF) sends no nulls and no rubouts.

3.1.3 NULLS AND LEADER

The Leader portion of a data-transmission tape may be of any length and may contain all characters except the start code of the format in use.

A tape to be input to the programmer may contain any number of null characters at the beginning or end. If data is output from the System 19 to a tape punch, the tape will contain a leader of fifty null characters, data in the specified format, and then fifty more null characters.

The programmer must send enough nulls after a carriage return to prevent certain terminals from missing characters. The Set Nulls Command provides the programmer with this pause capability by allowing sufficient nulls for the terminal to recover after each carriage return. The appropriate Select Code and procedure for setting nulls can be found in Section 3 of the System 19 O & M Manual.

3.1.4 START AND END CODES

Each translation format has a start code. Input to the programmer is ignored until the start code is received. On output the start code is sent immediately after the 50 nulls of leader.

NOTE

Data in ASCII Binary formats may also be input to or output from the programmer without start codes. See paragraphs 2.3.1 and 2.4.

All formats except Binary (without an arrow-shaped header) and DEC Binary transfer have an end code or an End-of-File record. The end code or record terminates input operations. (In ASCII-Hex and Octal formats, input will continue if a new start code follows within 16 characters of the end code.) Input can also be interrupted by the end of RAM or by depressing the KEYBD key on the System 19. Output data continues up to the last address in RAM. Binary (without an arrow-shaped header) and DEC Binary transfers require setting Block Limit L2 to indicate the end of transmission.

NOTE

When inputting data to the programmer, the sending machine will continue after transmission is complete unless it is equipped with the X-ON/X-OFF feature or controlled by the programmer's RTS line unless the number of bytes (Block Limit (L2) is specified. (See paragraphs 3.2.3-3.2.4).

3.1.5 ERROR CONTROLS

The System 19 incorporates certain routines for detecting errors in incoming data. Errors and Error Codes are defined in Appendix 1 of the System 19 O & M manual.

3.2 ADDRESS CONTROL

The System 19 uses 2 functions to control addresses in I/O operations. These are Block Limits and the address offset.

Block Limit L1, the Begin RAM Address. The value of L1 determines the first RAM address to or from which data will be transferred.

Block Limit L2, the Block Size. The value of L2 determines the number of data bytes that will be input or output.

Block Limit L3. This parameter is not used in I/O operations.

Address Offset. The address offset is used to adjust programmer-RAM addresses to the address range of larger memories. The address offset is subtracted from all addresses input to the programmer and added to all addresses output from the programmer.

Parameters L1, L2 and the address offset each have a default value which will be in effect if you do not specify a value for that parameter. These default values are listed in Table 3-2.

Table 3-2. Default Values for Address-Control Parameters

Parameter	Default Value
Block Limit L1	RAM address Ø
Block Limit L2	RAM limit less L1 for I/O operations
Address offset for input operations	First input address
Address offset for output operations	Ø

The Block Limits and address offset interact in slightly different ways, depending on whether data is being input to or output from the programmer, and whether the data stream includes address expressions.

Paragraphs 3.2.1 through 3.2.4 describe this interaction for each of the 4 possible combinations.

3.2.1 INPUT DATA WITH ADDRESSES

The address offset may be specified or left to default for input data with addresses. If the Begin RAM Address is

also specified, the difference between the first input address and the address offset will be added to the Begin RAM Address to determine the programmer RAM location for the first data byte. See Figures 3-1 and 3-2. Subsequent bytes are stored at successively higher RAM addresses. This relationship can be expressed algebraically:

$A_S = L1 + (A_I - A0)$

where

A_S = RAM address where the first data bytes will be stored

L1 = Begin RAM Address

A_I = input address of first data byte

A0 = address offset

An address offset can be specified which is greater than the first input address. The programmer will ignore all addresses below the value of the address. Thus, the programmer can "hunt" for a particular address in an input source, such as a paper tape, which contains lower addresses than the one sought. See Figure 3-3.

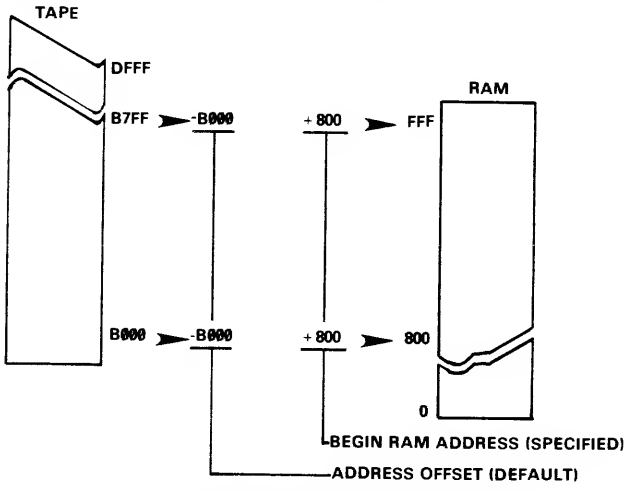


Figure 3-1. Setting the Begin RAM Address

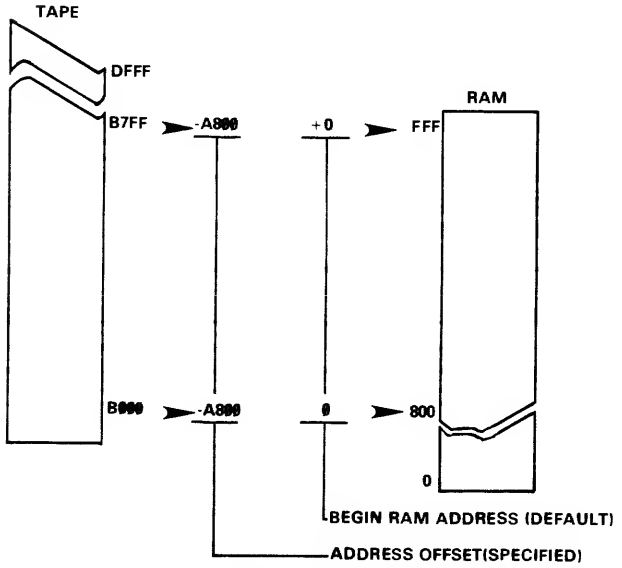


Figure 3-2. Setting the Address Offset

3.2.2 INPUT DATA WITHOUT ADDRESSES

The Begin RAM Address (Block Limit L1) determines where incoming data without addresses will be stored. L2 determines how many bytes the programmer will accept.

NOTE

The Binary (without an arrow-shaped leader) and DEC Binary formats do not use end codes. You must set Block Limit L2 when using these formats, in order to signal the end of an Input operation.

The first incoming byte of data will be stored at the Begin RAM address; succeeding bytes will be stored in sequential RAM addresses. If L1 is allowed to default, data storage in RAM will begin at address 0.

3.2.3 OUTPUT DATA WITH ADDRESSES

When data is output in formats with addresses, the Begin RAM Address (L1) determines where in RAM the output data will begin, and the address offset determines what value will be added to each outgoing address. Setting Block Limit L2 specifies the number of bytes that will be output.

For example, setting L1 to 800 and the address offset to C000 means that data will be output from RAM beginning at RAM address 800 and the first outgoing address will be equal to the address offset, C000. See Figure 3-4.

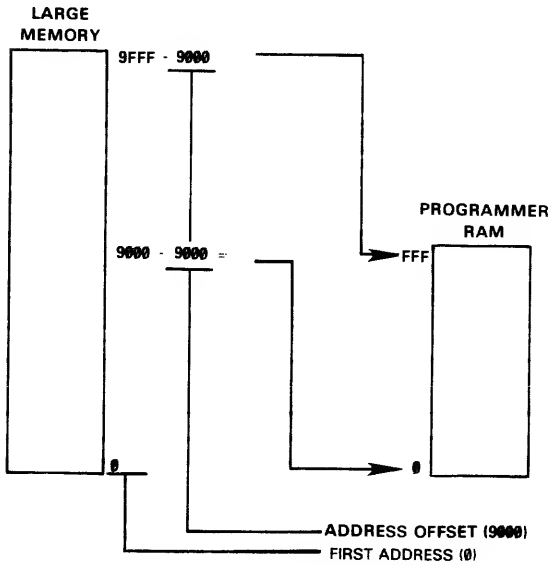


Figure 3-3. Address Offset Greater than First Input Address

3.2.4 OUTPUT DATA WITHOUT ADDRESSES

When data is output in formats without addresses, the Begin RAM Address determines where in RAM the output data will begin; the address offset is invalid. Setting Block Limit L2 will specify the number of bytes that will be output.

For example, setting the Begin RAM Address to 600 means that data will be output from RAM beginning at address 600.

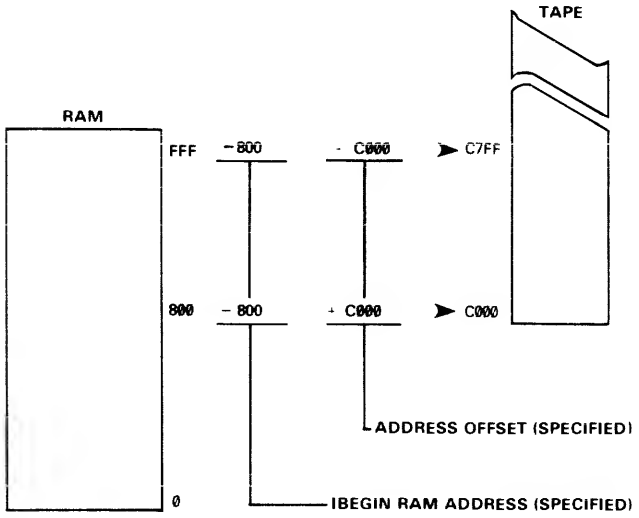


Figure 3-4. Output Addressing with Address Offset and Begin RAM Address.

3.3 TRANSMISSION PROCEDURES

3.3.1 SUMMARY

For inputting data, you must specify the operation, while format and address controls may be specified or allowed to default.

For outputting data, the operation must be specified, while address controls, record size, format and the number of nulls following the data can be specified or allowed to remain at their default value.

During an Input operation data is input to the programmer, which automatically checks for proper form and content (except in DEC Binary or Binary without an arrow-shaped header) and translates and stores the input data one byte at a time in RAM. With the input data resident in RAM, the operator may edit the data as desired prior to programming operations. Input to the programmer is initiated by a Select Code Command.

Data integrity can be established using the Input Compare operation. In this operation, incoming data is verified against stored RAM data, and the programmer displays an error message in case of a discrepancy. The Input Compare operation is initiated by the proper Select Code Command.

Data stored in RAM may be output upon command. Programmer bytes are translated to ASCII codes, except Binary and DEC Binary, which are in a form suitable for hard copy or for data storage and subsequent re-entry. Output is initiated by the Output Command.

3.3.2 KEYSTROKE FLOWCHART

Figure 3-5 is a flowchart of keystrokes used to initiate an Input, Input Compare or Output operation for data with or without addresses.

Set-up commands, such as BLOCK LIMITS (L1 and L2 only) and the translation-format Select Code are entered first. Output operations allow setting the number of nulls to be output following a carriage return, and some formats allow setting the number of bytes per output record. These parameters are entered next.

The operation Select Code (D0 - D6) is entered next. An address offset, if desired, is entered immediately after the operation Select Code.

When the operation is complete, observe the sum-check in the address display. This figure can be used to verify subsequent operations involving the same block of data.

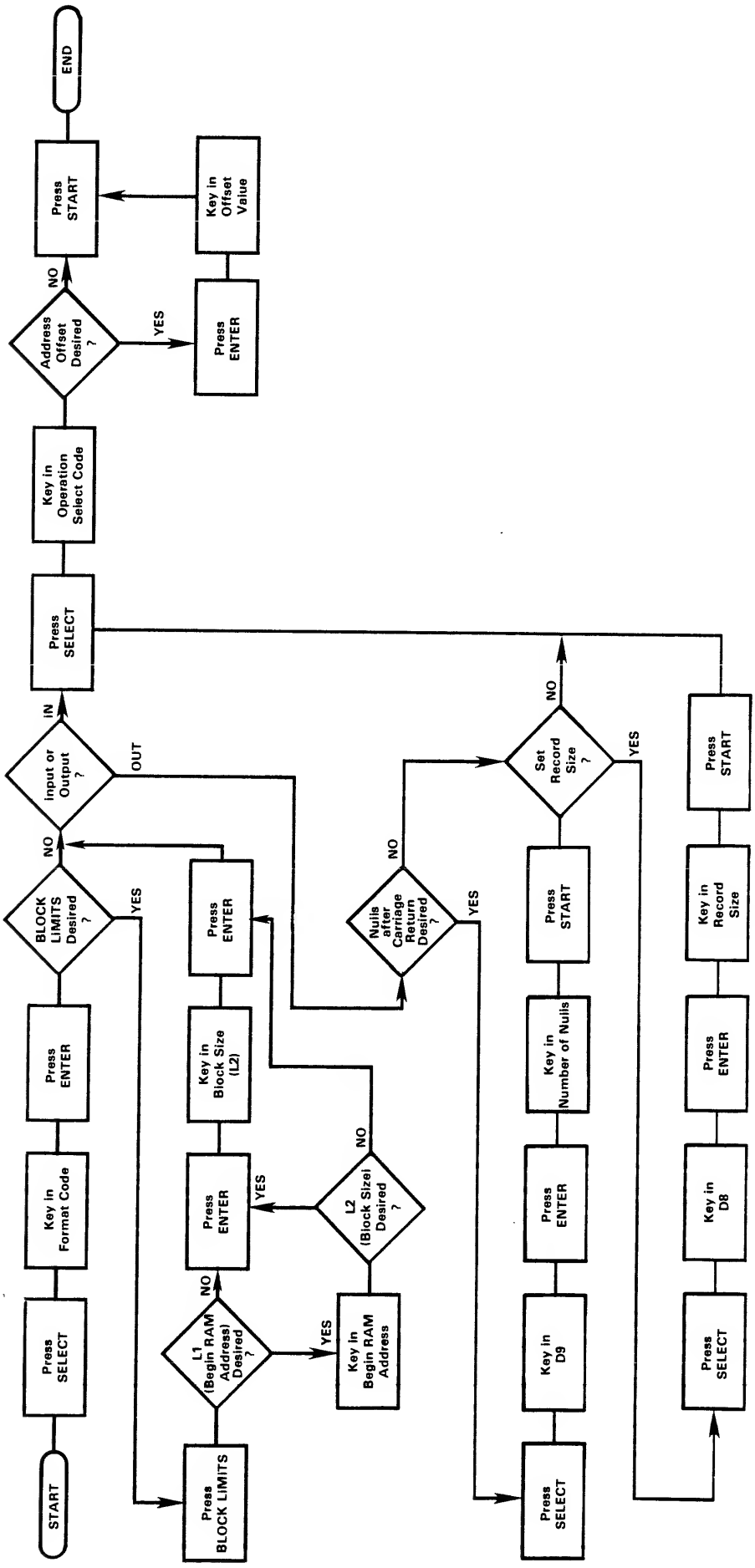


Figure 3-5. Keystrokes for I/O operations.